Optimisation of Message Distribution in Ad-hoc networks

Tomáš Bača*
Department of Software Technologies
Faculty of Management Science and Informatics
University of Žilina
Univerzitná 8215/1, 010 26 Žilina
tomas.baca@kst.uniza.sk

Abstract

Vehicular Ad hoc network (VANET) can serve as a source and storage of up-to-date data that improve safety, fluency and comfort in road traffic. We aimed in this dissertation thesis on improvements in end-to-end communication between two VANET nodes by means of proposing a new application-level communication protocol that is specialised on transferring large object structures. In order to minimise size of transferred message, the protocol makes use of the information about the structure that both communicating parties already share. The protocol uses UDP as an end-to-end datagram transportation service, because it is, according to our simulation results, more efficient than TCP. In order not to waste successfully received datagrams, the proposed protocol uses addressing that allows processing of independently received datagrams and using their contents for reconstructing the transferred message. In case of not receiving all the datagrams the message cannot be reconstructed completely, its missing parts are exactly denoted so that a receiving application is able to make use of at least the received part.

Categories and Subject Descriptors

E.4 [**Coding and Information Theory**]: Data compaction and compression

Keywords

ad hoc, VANET, TCP, UDP, data encoding, AdHocSim.FRI

*Recommended by thesis supervisor: Associate Prof. Emil Kršák. Defended at Faculty of Management Science and Informatics, University of Žilina on August 23, 2012.

Bača, T. Optimisation of Message Distribution in Ad-hoc Networks. Information Sciences and Technologies Bulletin of the ACM Slovakia, Vol. 4, No. 4 (2012) 1-6

1. Introduction

Vehicular Ad hoc Network (*VANET*) is wireless network with nodes communicating in peer to peer manner. Most of the network nodes are inside cars that are driving according to road traffic rules, although several nodes are inside elements of road traffic infrastructure, e.g. intelligent traffic sign or parking area. According to Lisbon agenda[1] *VANET* should contribute to road traffic safety by transferring messages about immediate danger. Secondly, improvements in road traffic fluency can be achieved by assisting drivers in picking optimal path or in catching green wave. Thirdly, user applications can communicate via VANET and improve passengers' comfort [10, 13, 14].

Our research is focused on development of user applications adjusted to ad hoc networks with rapidly changing topology. Especially, we focus on data dissemination and distribution through *VANET*. Two years ago a colleague of ours proposed architecture and principles of operation of distributed management system for mobile and vehicular ad hoc networks in his dissertation thesis. This year another colleague proposed and compared various algorithms for data replication via *VANET* in his dissertation thesis. However both these thesis are based on assumption that they can use communication layer with specific features and such a communication layer is proposed in this dissertation thesis [3, 4].

2. Issues with Transferring Messages via VANET

Most scientific papers, which deal with *VANET*, focus on improvements at the first three layers of *ISO/OSI* model – Physical, Data Link and Network Layer. Other papers seek mainly passenger safety and improvements in fluency of road traffic [5]. However there is still certain amount of communication capacity in *VANET*, which seems to be mostly unnoticed. We believe that this "unnoticed" capacity can be used for passengers' comfort by specialised user applications with proper adjustments in their architecture.

Yet we have identified one considerable issue that is likely to be encountered by developers of user applications for *VANET*. Delivery of large messages via *VANET* is a nontrivial task, although it might be often required, for example by distributed database system *ad-db .FRI*. We have based our assumption on these reasons:

[1]Plan to make European Union the most competitive and dynamic knowledge-driven economy by 2010

- Due to high mobility of network nodes, topology of the whole network is changing rapidly. Therefore every established connection between two network nodes is likely to be interrupted in a short time.
- Communication of user applications is transported via *VANET* with lower priority than safety and security messages therefore their transmission is likely to be postponed.
- According to various researches [2, 7, 9], communication via mobile and vehicular ad hoc networks suffers from increased ratio of undelivered packets compared to networks with fixed topology.

Therefore we do not consider traditional method, which would use any of common data encoding[2] and transport the data via *TCP* stream [12] well-suited for *VANET*. We assume that the communication would be possible but inefficient because of these reasons:

- *TCP* is calibrated for wired networks. It assumes that packet loss means traffic congestion and responds with decreasing communication speed. However in case of undelivered packets due to temporary interference the speed reduction would not be appropriate action.
- *TCP* suffers by so called "head of line blocking" problem, which should be expected to occur in *VANET* much more often than in wired networks.
- In *VANET* it is important to not waste limited period of time for message transfer. Therefore wireless communication standard *IEEE 802.11p*, used in *VANET*, avoids communication establishing handshakes, which are otherwise common in *IEEE 802.11* family of standards [1, 6, 10]. We think that *TCP* "three-way-handshake" should be avoided as well.

We have considered replacing *TCP* by *UDP* [11], but *UDP* itself would not suffice, because it does not guarantee in-order delivery of datagrams nor data completeness. Message, encoded by well-known methods, cannot be transported in this manner, because loss of arbitrary part would render the whole message invalid and received parts useless. Therefore we intend to propose a more efficient communication protocol that would perform better in *VANET*.

3. Goal of the Thesis

Main goal of the dissertation thesis is proposition of a communication protocol that would maximise amount of useful[3] data that can be transferred from one network node to another via *VANET* environment. The proposed protocol has to conform to the following conditions:

- The protocol is expected to be able to transfer relatively large object or data structures, such as database requests and responses.
- Due to high probability of network errors during message transfers, the protocol must include capability of recovering from errors.
- Due to high probability of packet loss and interruption of message transfer, the protocol must be able to extract useful data from received message, even if there are some parts of the message missing.
- In case of missing parts of the transferred message, the protocol must unmistakably distinguish lost parts of transferred object or data structure. It is not required to retrieve missing parts from the sender, because it might not be possible.
- Due to the fact that the network capacity is more valuable than local computing resources of network nodes, the protocol must minimise size of a message even if it meant higher demands on computing and memory resources of individual nodes.

4. Comparison of Transport Layer Protocols

According to the standard, duplex communication in *VANET* must use *IPv6* protocol as networking service. While designing a new application layer protocol, we have to therefore firstly decide, which one of the transport layer protocols[4] should be used atop the IP. According to the standard *vanetmodel* there are two options *TCP* and *UDP*.

Both *TCP* and *UDP* have their special features that designate them for uses in different situations. We have tested them both with the same task to learn how they are suited for transporting large amount of data through *VANET*. Network nodes were sending periodic requests, which were responded by their momentary neighbours with oversized responses. We have been comparing amount of useful data that were transferred inside responses and amount of data that had to be used for maintaining communication.

4.1 Verification Method

Under ideal conditions we would have deployed testing applications to real on-board-units capable of wireless ad hoc communication. However computer simulation is much more often used method for performing experiments with mobile and vehicular ad hoc networks [8] and it proved to be less demanding on human and technology resources in our case as well. During our research we have tried several renowned simulation tools[5] but with little success. We had issues with licenses, price and capabilities of the tools. Therefore we have decided to create our own simulation tool that would fit our needs and anticipations.

We have called the simulation tool *AdHocSim.FRI*. It is a discrete event simulator targeted at end-user applications in *VANET* environment. We have already used it in several experiments and published obtained results at international conferences.

4.2 Description of the Experiment

We have used AdHocSim.FRI to create model, which was composed of separate sub-models:

[2] E.g. ASN.1 DER, XML, JSON, YAML

[3] Usefulness of data is considered from the end-user point of view. Everything, except object or data structure that user intends to transfer, is not considered useful data, although its transfer is usually also necessary.

[4] Protocols of the 4th layer of ISO/OSI network model.

[5] We tried QualNet, NCTUns6.0, EstiNet and studied NS-2, NS-3 and IT Guru.

- Model of road network situated within urban area. It covered area approximately 4 km long and 2 km wide with rectangular net of roads. Total length of roads counted approximately 100km.
- Model of road traffic composed of individual cars. We used Poisson process to generate new cars entering the road network at borders of the simulated area. Every car had chosen its own path via road network, which it followed and eventually left the simulated area. Travel speed of cars ranged in between 10 m/s to 20m/s. Due to randomness of this model the actual number of cars fluctuated around average which was approximately 335 cars.
- Model of network communication. It has separate implementation for every main protocol used at lower four layers of ISO/OSI model.
  - Line-of-sight evaluation, Friis transmission equation, evaluation of mutual signal cancelation among distinct cars and signal-to-noise ratio evaluation had been used in calculation of probability, that a receiver receives transmitted message correctly.
  - Model of *IEEE 802.11p* had been used for signalling and collision avoidance.
  - Model of *IPv6* had been used for routing IP packets, although only single-hop transfers had been used in the simulation.
  - Model of *TCP* had been used for reliable byte-stream oriented communication between end-user applications.
  - Model of *UDP* had been used for unreliable datagram oriented communication between end-user applications.
- Model of transmission control mechanism for *UDP*. It is a well-known fact that *UDP* does not provide any feedback to the sender of datagrams nor prevents overloading of the network. We solved this issue by implementing simple mechanism described in following steps:
  - When a node received datagram with database response, it reacted by transmitting short datagram to *broadcast* address, saying that it is still receiving responses. Then it always waited at least for 50 milliseconds till broadcasting another short datagram.
  - A node, that was sending database response, always waited at least 4 milliseconds between transmitting two datagrams of the message.
  - If a node, that was sending database response, had not received at least one datagram from the node to which the message had been addressed during last 3 seconds, the connection had been considered broken and the transmission of the message had been terminated.

The experiment consisted of 20 independent simulation runs for each of the tested protocols. Every simulation run consisted of 1230-second-long warm-up period and 570-second-long evaluation period.

4.3 Results of the Experiment

During the simulation a separate record about individual database requests and database responses had been created and later on records that would compromise statistical information with invalid data were discarded. We were afterwards able to extract several various indicators; however we consider the following ones the most important:

- Amount of useful data, which were successfully received and extracted from database response.
- Amount of all data that were transmitted during the database response transfer. Besides successfully received data there are also included headers of *IP* packets, *UDP* datagrams, *TCP* segments and also all datagrams and segments that were used for transmission initiation and management and also retransmission of lost *TCP* segments.
- Efficiency of database response transfer computed by the following formula:

$$\text{efficiency} = \frac{\text{amount of received useful data}}{\text{amount of all transmitted data}}$$

By aggregating records of individual data transfers we have obtained information that is summarised in the Table 1.

	TCP	UDP
Average amount of received useful data per database response	1 MB	1,25 MB
Portion of transfers with amount of received useful data lower than 250KB	45%	32%
Portion of transfers with zero efficiency[6]	18%	7,5%
Average efficiency of data transfer	52%	76%
Maximal efficiency of an individual transfer	95%	97%

Table 1: Results of the experiment

According to the results, we have concluded that it is possible to transfer more useful data with less overhead if transported in *UDP* datagrams than in *TCP* stream. There is also lower probability of zero efficiency communication, or in other words, higher probability that at least some part of the message would have been transferred successfully. We have therefore decided to use *UDP* as a transport layer protocol and to base proposed protocol upon it.

5. Proposed Protocol

In the previous section we have explained that we intended to use *UDP* as a transportation service for endpoint-to-endpoint communication. Therefore we had to take features of *UDP* into consideration while designing proposed communication protocol. Namely the fact

[6] Sender of the message initiated its transfer but receiver was unable to receive any byte of useful data.

that *UDP* transfers datagrams, which travel through *IP* network independently of each other and that sender of a datagram gets no feedback on whether the datagram reached its destination or not. Likewise the receiver is not hinted by means of *UDP* whether any datagram had been lost, duplicated or received out-of-order during the transmission.

Therefore, we can expand the list in Section 3 by the following requirement that the proposed protocol must satisfy:

- Due to the fact that *UDP* transfers datagrams with limited size[7] , which is considerable less than expected size of transferred messages, the protocol must be able to split any message to datagrams with defined size limit.
- Due to the fact that *UDP* datagrams are transported independently of each other, there must be enough information in every datagram so that it is possible to process it independently of other datagrams being received or not.

5.1 Properties of Transferred Object Structure

The proposed protocol is designed for transferring object structures, which can be seen as a composition of objects with one major object that directly or indirectly references every other object. Object structures often resemble tree with the major object being the root of the tree. In case of reference cycles within the structure, these cycles can be decomposed according to rules explained in section 5.3. Every transferred object structure is therefore assumed to be a tree with root that directly or indirectly references every other object.

It is also expected that every node in the tree structure carries enough information within itself to be at least partially meaningful without values of its child nodes however its data are not useful without data stored in its parent node. This expectation is based on assumption that elementary values (e.g. numbers, texts, etc.) are stored in leaf nodes, because they do not reference any other objects. Non-leaf objects are complex and are responsible for creating multi-level tree structure because they are composed mainly of references to other objects.

By obeying explained rule and if every received datagram of the message should be useful regardless of other datagrams being received, it has to carry data from root node and every intermediate node that are between the root and leaf nodes. However, it is not necessary to write any intermediate node data to the datagram if there is not any data from its child nodes. In other words, the protocol should write data from leaf nodes to datagrams and append parent node for every written node.

This rule may appear to cause a lot of redundancy because of repeating data from non-leaf nodes to every datagram. Yet we believe that most of non-leaf nodes do not carry any important data that should be transferred, because they are already known to both sender and receiver of the message.

5.2 Declaration of Object Structure

Both sender and receiver of the message are expected to share certain amount of common knowledge about the transferred object structure. Namely, both of them are required to provide the same declaration of the transferred object structure. Therefore classes of transferred objects are usually obvious, although the declaration might not be the same for every message that is transferred. For that reason, the sender has to add certain identifier of the declaration to the message that the receiver would be able to comprehend. Structure and contents of the identifier is solely application dependent.

For example if a distributed database system were used in the network, all the nodes would use the same global conceptual schema of the database. In this scenario, the same declaration of the database response should be derived from one database request by any node. Therefore if a responding node put identifier of the database request to the database response, both it and requesting node would process the encoded message the same way.

There are three basic types of object classes recognized by the proposed protocol:

- Composition of other objects identified by named attributes. Declaration of such class contains fixed list of attributes and classes of attributes. Object of this type of class does not store any other data that would not be obvious from the declaration and should be therefore transferred.
- Collection of objects, usually of the same class. Elements of the collection are identified by their sequence numbers. Declaration however does not provide any information on the total number of the elements. It is therefore important to transfer length of the collection within the message.
- Elementary value, which might be numerical, logical or textual. Its class and required size in bytes is usually obvious from declaration except for textual values, which usually varies in length.

There are cases, when class of an object might not be obvious. It happens when the object interface is used in declaration instead of class or when a class that is subclassed is declared. In that case declaration gives several options on classes of a certain object within the structure and identification of the object class must be included to the transferred data.

5.3 Decomposition of Object Structure to a Tree

We have already mentioned that object structures might include reference loops. Loops are necessary, if two or more objects should share the same data, either for mutual communication or for saving memory. It is however preferable to transfer tree structures, thence we propose loop decomposition by extending the object structure with new objects of class *Address*.

For every object in transferred structure, which is referenced more than once, only one reference would remain.

[7]Size of the *UDP* datagram is limited by packet size of the underlying IP protocol, however for performance reasons we would rather to conform also to frame size of data-link layer protocol, so that one datagram does not have to be fragmented across multiple frames.

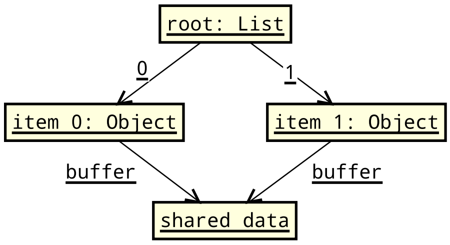

Figure 1: Object structure with reference cycle

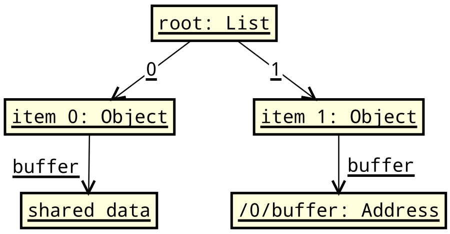

Figure 2: Object structure after cycle decomposition

Other references would be replaced by Address objects, which store absolute path from root node to the object. See the example at figures 1 and 2.

5.4 Encoding Object Structure into Datagrams

After decomposing reference cycles, pre-order traversal of the tree structure is used to pass every node of the tree. For every node, that is passed, necessary data are encoded and temporarily stored in a memory buffer with the same size as size of datagram payload. In addition to data from nodes, identifier of the message must be stored in every datagram and is therefore encoded to the beginning of the buffer.

The traversal is interrupted after encoding data from the last leaf node of the tree that can be stored in the buffer without overrunning its capacity. The buffer contents are sent in payload of a datagram and then the buffer is emptied. Then the pre-order traversal is resumed, however it follows the rule that in every datagram there must be data from parent node of any node of which data are stored in it. Therefore the traversal is resumed from the root node and follows to the first leaf node that hasnâĂŹt been encoded yet.

It is clear that by complying with the described rules, every datagram starts with the root node followed by one of its child node and so on till one of leaf nodes is reached and afterwards the tree is pre-order traversed. If the datagram is to be processed independently of other ones, the picked path from the root node to the first leaf node must be denoted by addressing the picked child node.

For every encoded node the following data are written to the datagram:

1. Code of the object class, if it is not obvious from the declaration.
2. Size of the object, if it is collection or text value or its size is not obvious from the declaration.
3. Value of the object, if it is leaf node.
4. Relative path to the next node, if it is not obvious from pre-order traversal rule.

5.5 Receiving the Message

A receiver of the message is supposed to process and make use of every received datagram regardless of having received other datagrams belonging to the same message as well. Using the identifier of the message, carried by every datagram, the receiver can derive declaration of the transferred message from whichever datagram is received first and join contents of all received datagrams together. By including complete path from root node to leaf nodes in every datagram, the receiver does not need contents of any datagram to be able to restore part of the object structure carried by another datagram.

If some datagrams were not received, contents of leaf node objects would not be replaced nor derived from redundancy of any sort. It is also possible that contents of several non-leaf node objects would not be received as well. In that situation, the received object structure would not be complete. However, the declaration and received parts of the message provide enough information to reconstruct at least parts of the object structure. It is also possible to exactly denote missing parts with specialised objects with special "unknown" value.

6. Conclusion and Future Work

We have achieved the main goal of the dissertation thesis, to maximise amount of transferred useful data via *VANET*, by proposing a new communication protocol. Several techniques are included in the protocol to accomplish the goal.

Messages do not carry any portion of data, that both sender and receiver are able to derive from information both of them already share. Compared to *ASN.1 DER* messages are reduced in size approximately by one third[8].

Messages are transported in *UDP* datagrams instead of commonly used *TCP* stream, which we have proved to be more efficient way of communication for *VANET* environment. Inner structure of every message fragment, transported in separate datagram, allows for independent processing by receiver of the message. The receiver is therefore able to extract useful data from every received datagram no matter how many datagrams were lost during the transmission. Missing parts would not render the rest of the received message unusable, instead of that the message would have its missing parts explicitly denoted.

There are however limitations in use of the proposed protocol, although they rather result from the environment and the protocol adapts to them. Communication in *VANET* in general is prone to interruptions, which are likely to cause incompleteness in receiving messages. It is the fact that should be taken into consideration by anyone who is designing applications for *VANET*.

In any case the proposed protocol can still be improved. In our future research we would like to modify it to use

[8]The exact ratio depends on portion of knowledge both sender and receiver share and the message itself.

transportation protocol *SCTP* instead of *UDP* and compare their performance. Moreover we suppose that the protocol could benefit from ability to retransmit missing datagrams, if the communication parties still had time to communicate.

Acknowledgements. This contribution/publication is the result of the project implementation:

Centre of excellence for systems and services of intelligent transport II., ITMS 26220120050 supported by the Research & Development Operational Programme funded by the ERDF.

„Podporujeme výskumné aktivity na Slovensku. Projekt je spolufinancovaný zo zdrojov EÚ“

References

[1] R. and Inovative Technology administration. Its standards fact sheets. online, september 2009. navštívená v apríli 2012.

[2] M. Fazio, A. Biundo, M. Villari, and A. Puliafito. An in-depth analysis of the sctp behavior in mobile ad hoc networks. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops, 2007. WiOpt 2007. 5th International Symposium on*, pages 1 – 9, apríl 2007.

[3] J. Janech. Distributed database systems in the dynamic networks environment. *International Journal on Informatipn Technologies and Security*, (1):43–52, 2010.

[4] J. Janech. *Riadenie procesov pri distribúcii databáz*. PhD thesis, Žilinská Univerzita v Žiline, Fakulta riadenia a informatiky, Žilina, 2010.

[5] M. Jerbi, S.-M. Senouci, R. Meraihi, and Y. Ghamri-Doudane. An improved vehicular ad hoc routing protocol for city environments. In *Communications, 2007. ICC '07. IEEE International Conference on*, pages 3972 –3979, june 2007.

[6] D. Jiang and L. Delgrossi. Ieee 802.11p: Towards an international standard for wireless access in vehicular environments. *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, pages 2036 –2040, may 2008.

[7] H. Lim, K. Xu, and M. Gerla. Tcp performance over multipath routing in mobile ad hoc networks. In *Communications, 2003. ICC '03. IEEE International Conference on*, volume 2, pages 1064 – 1068 vol.2, máj 2003.

[8] D. A. Maltz, J. Broch, and D. B. Johnson. Experiences designing and building a multi-hop wireless ad hoc network testbed. Technical report, 1999.

[9] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand. Achieving 100% throughput in an input-queued switch. *IEEE Transactions on Communications*, 47(8):1260–1267, august 1999.

[10] R. Popescu-Zeletin, I. Radusch, and M. A. Rigani. *Vehicular-2-X Communication*. Springer-Verlag, Berlin Heidelberg, 2010.

[11] RFC0768. User datagram protocol, august 1980.

[12] RFC0793. Transmission control protocol, september 1981.

[13] P. E. únie. Deployment of intelligent transport systems in europe, 2010. prístup 10-apríl-2010.

[14] P. E. únie. Europa – glossary – lisbon strategy, 2010. navštívená v apríli 2010.

Selected Papers by the Author

T. Bača. Benefits of coroutines in discrete-event simulation. In: *Objekty 2011 : proceedings of the 16th international conference on object-oriented technologies*. Žilina, 2011.

T. Bača, J. Janech, A. Lieskovský. Distribution of information through vanet in urban environment. In: *Distance learning, simulation and communication 2011 (DLSC 2011)*. Brno, Czech Republic, 2011.

T. Bača, E. Kršák. AdHocSim.FRI - simulator of vanet applications. In: *Journal of Information, Control and Management Systems*. 2011. vol.9, 3 Spec. iss.

T. Bača, A. Lieskovský. Data dissemination in VANET. In: *TRANSCOM 2011 : 9-th European conference of young research and scientific workers*. Žilna, June 2011. n°3.

J. Janech, T. Bača. Distributed database system in vehicular ad-hoc network environment. In: *Journal of Information, Control and Management Systems*. 2010. vol.8, n°2.

J. Janech, T. Bača. Distributed Database Systems in Vehicular Ad-hoc Network. In: *Communications*. Žilina, 2010.

J. Janech, T. Bača. Distributed database systems in the dynamic networks environment. In: *Communications : scientific letters of the University of Žilina*. 2011. vol.13, n°4.

J. Janech, T. Bača, M. Tavač. Distribuovaný databázový systém v prostredí ad-hoc sietí. In: *Datakon 2010*. Mikulov, Hotel Eliška, Czech Republic, October 2010.

A. Lieskovský, T. Bača. Dissemination of safety data in vehicular Ad hoc network. In: *QUAERE 2011: recenzovaný sborník příspěvků interdisciplinární mezinárodní vědecké konference doktorandů*. Hradec králové, Česká republika, April 2011.

A. Lieskovský, T. Bača, J. Janech. Comparison of data distribution in VANET urban environment. In: *Digital technologies 2010: 7th international workshop on digital technologies, circuits, systems and signal processing*. Žilina, Slovakia : EDIS, November 2010.

A. Lieskovský, J. Janech, T. Bača. Data replication in distributed database systems in VANET environment. In: *ITST 2011 : 11th international conference on ITS Telecommunications*. St. Petersburg, Russia, August 2011.

A. Lieskovský, J. Janech, T. Bača. Data replication in distributed database systems in VANET environment. In: *ICSESS 2011 : proceedings 2011 IEEE 2nd international conference on software engineering and service science*. Beijing, China, July 2011.